

WHAT IS CLAIMED IS:

1 1. A system for translating foreign binary code for execution on a host
2 computer system where the host computer system is architecturally distinct from the foreign
3 architecture, said host computer system providing a foreign and a host virtual space, said
4 computer system further comprising:
5 a first page table representative of the logical allocation of physical memory
6 associated with said foreign binary code;
7 a second page table representative of the logical allocation of physical
8 memory associated with binary translated code; and
9 means for maintaining said first and second page tables in response to changes
10 generated by executing said binary translated code.

1 2. The system of claim 1 wherein said maintaining means further
2 comprises a register dedicated to track code modification to maintain correspondence
3 between foreign and binary translated host code flag, associated with each page of memory,
4 to indicate modification of said page of memory.

1 3. The system of claim 1 further comprising a flag, associated with each
2 page of memory, to indicate a change in a page in foreign virtual space.

1 4. The system of claim 1 further comprising means for restricting access
2 to a page of memory in foreign virtual space.

1 5. The system of claim 4 wherein said restricting means comprises means
2 for generating a trap.

1 6. The system of claim 1 wherein said first page table maps foreign code
2 to a portion of physical memory.

1 7. The system of claim 6 wherein said first page table includes a first and
2 second bit associated with each page of memory in foreign virtual space, said first bit
3 indicating whether a page of memory in foreign virtual space is write protected and said
4 second bit indicating a locked page of memory

1 8. The system of claim 7 further comprising means for determining the
2 state of said first bit and invoking a page fault exception upon a change in page contents.

1 9. The system of claim 1 wherein said first page table includes a first and
2 second bit associated with each page of host memory, said first bit indicating whether access
3 to said page is restricted; said second bit indicated whether access to said page in said foreign
4 virtual space is accessible only in a supervisor mode of operation.

1 10. A method for translating foreign binary code for execution on a host
2 computer system where the host computer system is architecturally distinct from the foreign
3 architecture, said host computer system providing a foreign and a host virtual space, said
4 method comprising the steps of:

5 configuring a first and second virtual space;
6 storing foreign code in said first virtual space;
7 storing a plurality of translation processes in said second virtual space;
8 storing binary translated code in said second virtual space, said binary
9 translated code comprising at least a portion of said foreign code in said first
10 virtual space;
11 executing said binary translated code;
12 detecting a memory access that attempts to modify a memory location;
13 determining the status of the accessed memory location;
14 if the status permits access, modifying the memory location;
15 if the status does not permit access, generating a page fault exception.

1 11. The method of claim 10 further comprising the step of:

2 if the status indicates access is limited to code executing in an emulated target
3 supervisor mode, determining if the computer system is executing in an emulated target
4 supervisor mode; and

5 if the status indicates that the code is not executing in said emulated target
6 supervisor mode, generating an exception.

1 12. The method of claim 11 further comprising the steps

2 if access is allowed, checking the address against a range of addresses; and
3 determining if the memory location to be accessed is currently in memory.

1 13. A method for translating foreign binary code for execution on a host
2 computer system where the host computer system is architecturally distinct from the foreign
3 architecture, said host computer system providing a foreign and a host virtual space, said
4 method comprising the steps of:

5 configuring a first and second virtual space;

6 storing foreign code and data in said first virtual space;
7 storing binary translated code in said second virtual space, said binary
8 translated code comprising at least a portion of said foreign code in said first virtual space;
9 defining a first page table to map said foreign code from said first virtual space
10 to physical memory;
11 defining a second page table to map said binary translated code to said second
12 virtual space;
13 associating information relating to said foreign code with said page table, said
14 information determining if a portion of said foreign code and data in said virtual space may
15 be accessed.

1 14. The method of claim 13 further comprising the step of using said
2 information to determine if said foreign code comprises operating foreign system code.

3 15. The method of claim 14 further comprising the step of using said
4 information to detect modification of said code and data in said foreign virtual space.

5 16. The method of claim 15 further comprising the step of invoking a
6 binary translation process upon detection of modification of said code and data in said foreign
7 virtual space to generate binary translated code corresponding to said code and data in said
8 foreign virtual space.

9 17. The method of claim 13 further comprising the step of using said first
10 page table to map a logical address of said foreign code to a compatibility region of physical
11 memory in said host computer system.

1 18. A method for translating foreign binary code for execution on a host
2 computer system where the host computer system is architecturally distinct from the foreign
3 architecture, said host computer system providing a foreign and a host virtual space, said
4 method comprising the steps of:

5 configuring a first and second virtual space;
6 storing foreign code and data in said first virtual space;
7 storing binary translated code in said second virtual space, said binary
8 translated code comprising at least a portion of said foreign code in said first virtual space;
9 defining a first page table to map said foreign code from said first virtual space
10 to physical memory;

11 defining a second page table to map said binary translated code to said second
12 virtual space;
13 providing a support software layer, said software layer functioning as an
14 operating system for controlling the binary translation of said foreign code and data and for
15 detecting a write access to physical memory of said host computer system, said physical
16 memory corresponding to said foreign virtual space by said host code.

1 19. The method of claim 18 further comprising the step of executing said
2 support software layer in said second virtual memory, said support software layer adapted to
3 managing I/O operations.

1 20. The method of claim 19, further comprising the step of virtualizing a
2 restricted set of functions.

1 21. The system of claim 20, further comprising the step of virtualizing
2 peripheral components expected in a target operating system.

1 22. A system for translating foreign binary code for execution on a host
2 computer system where the host computer system is architecturally distinct from the foreign
3 architecture, said host computer system providing a foreign and a host virtual space, said
4 computer system further comprising:

5 a first page table representative of the logical allocation of physical memory
6 associated with said foreign binary code;

7 a second page table representative of the logical allocation of physical
8 memory associated with binary translated code; and

9 a support software layer adapted to maintain correspondence between foreign
10 code and data in said foreign virtual space and a set of host instructions in said host virtual
11 space.

1 23. The system of claim 22 wherein said support software layer comprises
2 means for performing binary translation of said foreign code and said host code.

1 24. The system of claim 23 further comprising an access watch engine
2 adapted to detect a write access to physical memory in said host computer system.

1 25. The system of claim 22 further comprising means for detecting write
2 accesses to physical memory corresponding to said foreign virtual space.

1 26. The system of claim 25 further comprising means for detecting write
2 accesses to physical memory corresponding to memory mapped peripherals.

1 27. The system of claim 25 further comprising means for virtualizing a
2 restricted set of available hardware to produce enough resources to run resource-consuming
3 firmware.

1 28. A system for translating foreign binary code for execution on a host
2 computer system where the host computer system is architecturally distinct from the foreign
3 architecture, said host computer system providing a foreign and a host virtual space, said
4 computer system further comprising:

5 a physical memory;

6 a first page table representative of the logical allocation of physical memory
7 associated with said foreign binary code;

8 a second page table representative of the logical allocation of physical
9 memory associated with binary translated code;

10 a support software layer adapted to maintain correspondence between foreign
11 code and data in said foreign virtual space and a set of host instructions in said host virtual
12 space; and

13 an emulated supervisor flag associated with portions of said host binary code
14 where said support software layer accesses said supervisor flag prior to executing a write
15 access into said foreign and host virtual spaces.

1 29. The system of claim 28 further comprising means for detecting said
2 supervisor flag and marking selected portions of said foreign and host virtual spaces as
3 accessible only in an emulated supervisor mode, said detecting means coupled to said support
4 software layer.

1 30. The system of claim 28 further comprising a page lock flag, said page
2 lock flag indicating restricted access to selected portions of said foreign virtual space.

1 31. The system of claim 30 further comprising means for detecting the
2 state of said page lock flag.

1 32. The system of claim 28 further comprising a common structure
2 supported in hardware for maintaining said the foreign and host virtual spaces.

1 33. The system of claim 28 further comprising an access watch engine for
2 detecting write accesses to said physical memory.

1 34. The system of claim 28 further comprising an emulated target
2 supervisor flag for initiating binary translation from said foreign binary code to said host
3 binary code.